Effective Usage of Computational Trust Models in Rational Environments

LE-HUNG VU and KARL ABERER, Ecole Polytechnique Fédérale de Lausanne (EPFL)

Computational reputation-based trust models using statistical learning have been intensively studied for distributed systems where peers behave maliciously. However practical applications of such models in environments with both malicious and rational behaviors are still very little understood. In this article, we study the relation between their accuracy measures and their ability to enforce cooperation among participants and discourage selfish behaviors. We provide theoretical results that show the conditions under which cooperation emerges when using computational trust models with a given accuracy, and how cooperation can still be sustained while reducing the cost and accuracy of those models.

Specifically, we propose a peer selection protocol that uses a computational trust model as a dishonesty detector to filter out unfair ratings. We prove that such a model with reasonable misclassification error bound in identifying malicious ratings can effectively build trust and cooperation in the system, considering rationality of participants. These results reveal two interesting observations. First, the key to the success of a reputation system in a rational environment is not a sophisticated trust-learning mechanism, but an effective identity-management scheme to prevent whitewashing behaviors. Second, given an appropriate identity-management mechanism, a reputation-based trust model with a moderate accuracy bound can be used to effectively enforce cooperation in systems with both rational and malicious participants. As a result, in heterogeneous environments where peers use different algorithms to detect misbehavior of potential partners, cooperation may still emerge. We verify and extend these theoretical results to a variety of settings involving honest, malicious, and strategic players through extensive simulation. These results will enable a much more targeted, cost-effective and realistic design for decentralized trust management systems, such as needed for peer-to-peer, electronic commerce, or community systems.

Categories and Subject Descriptors: H.4.0 [**Information Systems Applications**]: General; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Intelligent agents*; *Multiagent systems*; I.6.3 [**Simulation and Modeling**]: Applications

General Terms: Algorithms, Economics, Design, Experimentation

Additional Key Words and Phrases: Trust, reputation, learning, rationality, incentive-compatibility

ACM Reference Format:
Vu, L.-H. and Aberer, K. 2011. Effective usage of computational trust models in rational environments. ACM Trans. Auton. Adapt. Syst. 6, 4, Article 24 (October 2011), 25 pages.
DOI = 10.1145/2019591.2019593 http://doi.acm.org/10.1145/2019591.2019593

1. INTRODUCTION

Reputation information has long been shown as an effective tool to enforce cooperation and build trust among participants in a variety of e-commerce systems and online

A short version of this article is published in *Proceedings of the Web Intelligence and Intelligent Agency Technology Conference (WI-IAT'08)*, 583–586.
Authors' address: L.-H. Vu and K. Aberer, Ecole Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland; email: {lehung.vu, karl.aberer}@epfl.ch.
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permission may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701, USA, fax +1 (212) 869-0481, or permissions@acm.org.
© 2011 ACM 1556-4665/2011/10-ART24 $10.00
DOI 10.1145/2019591.2019593 http://doi.acm.org/10.1145/2019591.2019593

forums such as eBay.com, Yahoo Auction (auctions.yahoo.com), or recommender systems (amazon.com). Due to the open nature of such applications, participants usually do not have incentives to cooperate, e.g., not to ship the items after receiving payments in an auction site, or not to contribute any resource. In those cases reputation based on user feedback can be an effective measure to isolate bad participants and promote good behaviors.

Existing reputation system designs can be generally classified into two classes, the first being the *computational trust models* that predict peers' behavior, e.g., whether a peer offers high quality services and gives reliable recommendations on the others, based on their historical performance. Such methods [Despotovic 2005; Anceaume and Ravoaja 2006; Sun et al. 2006] are designed to be resilient against different types of attacks from malicious users—those wanting to take the system down at any cost. Peers are assumed to behave probabilistically: those that have cooperated in the past are supposed to do similarly in forthcoming transactions. Computational trust models usually ignore the fact that many participants have economic incentives and behave rationally. The second solution category comprises the *mechanism design approaches* that use reputation as a sanctioning tool to enforce cooperation and establish trust among *rational* participants by penalizing bad peers and rewarding good ones [Dellarocas 2005a; Jurca and Faltings 2006; Miller et al. 2005]. Rational peers are opportunistic and adapt their behaviors strategically to maximize their expected lifetime utilities. The system is designed as a game whose structure ensures that being cooperative is the strategy in equilibriums of rational peers. Solutions in this class usually rely on many assumptions, such as peers being fully rational and having unlimited computational power to find the complex equilibriums of the mechanisms. More importantly, they do not consider the malicious behavior of attackers.

Since both rationally opportunistic and malicious behaviors can be present in real environments, an effective reputation management approach to minimize the influence of these unwanted behaviors is of paramount importance. A natural question is how we can exploit well-tested and accurate computational trust models to effectively establish trust among partners in environments where peers may exhibit various behaviors, including honest, malicious, and rationally opportunistic. More concretely, it is important to know under which conditions computational trust models that are resilient against malicious attackers can also motivate cooperation of rationally opportunistic players.

Another issue we want to address is to minimize the cost of using expensive computational trust models in environments where most peers are rational. A solution to this question implies many benefits for various applications, e.g., where peers have limited resources and quick decisions are generally preferred to avoid missing opportunities, especially in competitive scenarios. In fact, given the rationality of peers, it may be unnecessary to always use accurate yet costly learning algorithms to encourage truthful behaviors. The reason is that rationally selfish peers make use of their knowledge about the deployed algorithm(s) to avoid having bad reputations and maintain their high benefits from the system. Being aware of the existence of such learning algorithms that can reliably detect bad behaviors, peers have little incentives to cheat and thus expensive learning can be avoided. Consequently, accurate yet costly trust-learning mechanisms may be needed as an inspection tool to detect past cheating behaviors, in order to punish bad agents appropriately, e.g., by not selecting them for later transactions. Such punishment meted out to peers found cheating can be used to provide sufficient incentives for their cooperation.

This article proposes and analyzes a simple but effective way of using a computational trust model to minimize the influence of malicious peers and at the same time, keep rational peers cooperative during most of their transactions. Specifically,

we propose a peer selection protocol that consists of two steps. First, a computational trust model is used to estimate the reliability of the most recent rating on a peer. The result of this evaluation is then used to decide whether to include the target peer for selection or to ignore/blacklist the peer being rated. Thanks to its simplicity, the protocol can be easily applied in various open systems with different degrees of centralization. As a result, this work provides the following contributions.

— We prove that if the accuracy of the chosen computational trust model is sufficiently good, rational peers find it most beneficial to cooperate in all but some last transactions. According to extensive simulation, this result still holds if peers join and leave dynamically, provided that most of them stay long enough. The key to enforcing cooperation in such environments is an identity management scheme to effectively prevent whitewashing behavior, rather than the trust learning algorithm being used.
— We propose a cost-efficient trust management approach that ensures cooperation and trust in the system, while minimizing the related cost. Inspired by an inspection game-theoretic approach [Avenhaus et al. 2002], we prove that under certain assumptions, the evaluation of rating reliability can be done by using an accurate, yet expensive, computational trust model with a low frequency while still maintaining high cooperation in the system. As a result, the total implementation cost of the whole selection protocol can be significantly reduced.

As a theoretical contribution, this work bridges the gap between Friedman and Resnick [2001] and the existing works studying properties of various computational trust models. We show that besides the necessity of effective identity-management to avoid whitewashing behavior, in rational environments a small error bound in identifying malicious ratings by a computational trust model is sufficient to enforce cooperation. Therefore, any existing trust learning algorithm in the literature with a moderate error bound can be used to build cooperation among rational participants. Cooperation is generally possible in heterogeneous environments, where peers use different algorithms to learn the trustworthiness of their potential partners, as long as these algorithms guarantee a reasonable error bound.

Our system model will be presented in the next section. In Section 3 we analyze in detail the relation between the accuracy of a computational trust model and its ability to enforce cooperation of the rational participants. We then propose combining different trust models to minimize their usage cost while retaining the possibility of fostering cooperation among peers, in Section 4. Section 5 proposes an approximate approach to use computational trust models in dynamic scenarios with peers joining and leaving over time. Our simulation and experimental results are presented in Section 6. We summarize related work in Section 7 before concluding the article in Section 8.

2. SYSTEM MODEL

2.1. Example Applications

Throughout the article, the following example is used to illustrate the fundamental concepts and results of our work. We consider a decentralized (peer-to-peer) market of products or services, where each participant can be provider and/or client of certain services. As a concrete example, any person in the Internet can advertise and sell their goods or services in a peer-to-peer system or on an online social network like Facebook.com. Another realistic showcase of this is the recent launch of Neighborhoods (neighborhoods.ebay.com) that enables eBay users to do shopping via their social networks. Henceforth, we use the following notions of a client, a provider, or a service to illustrate our concepts in this example market of services. In other

application contexts these notions may refer, respectively, to a buyer, a seller, or a resource or an article to be sold in the system.

In this scenario, clients can search to buy the services that match their needs and interests. As a common rule, a client has to pay for the service first. Only after receiving the payment, the provider may provide the service with either low or high quality. E.g., in eBay a low quality service means either the seller cheats by not shipping an article, or the article is not of good quality as described. The traditional way to enforce the cooperation of a provider via a reputation mechanism is to allow a client to rate the service of the provider as good or bad after finishing the transaction [Resnick et al. 2000]. Other clients use the available ratings on a provider to decide whether they should do business with that provider. In this case, a computational trust mechanism is very helpful to effectively eliminate unreliable ratings and minimize the influence of malicious raters.

This article focuses on the cooperation among participants in the application layer: we assume the existence of a possibly decentralized storage system that supports efficient storage and search for publicly available information—namely the description of available services and their ratings. Such a storage system must ensure that advertisements of provided services, published ratings, and transaction information are authentic and cannot be tampered with. In other words, peers can neither fake transactions nor modify the feedback submitted by others. Although addressing these security goals are beyond the scope of our article, we believe existing solutions are available. For example, these goals can be met by using a DHT-based storage [Aberer et al. 2003] and cryptographic tools such as digital signatures based on a decentralized PKI infrastructure [Datta et al. 2003]. Also, centralized storage solutions, e.g., provided by *eBay*, can be considered sufficiently secure.

Peers are assumed to stay in the system long enough so that reputation information is effective. This assumption can be relaxed under certain circumstances, e.g., in centralized environments where it is possible to impose an entrance fee for a newly joined peer. In practice, this assumption also holds since identities are not cheap: users must invest time and money to build relationships before they can participate in business transactions with others. Therefore, it is better for most users to stay longer in the system rather than departing and starting everything all over with a new identity. The cheap identity issue is discussed in another work of ours [Vu and Aberer 2010].

These simplifying assumptions are standard and well-accepted in the trust and reputation research communities[Despotovic 2005]. More importantly, the system model is realistic—it is an abstraction of several practical P2P application scenarios with different degrees of centralization, where participants are rational to a certain extent. Such a scenario can represent, for example, a centralized eBay-like trading site, a social network-based system for selling and buying goods, or decentralized markets of computational or storage services [Buyya et al. 2001; Papazoglou and Georgakopoulos 2003]. Consequently, the proposed solution can be used in all of these applications.

2.2. System Model

Denote P the set of participants (peers) in the example P2P market of services. Let the mapping $W : P \times P \rightarrow D_f$ be the observable online social relationship among them, where D_f is the domain of relationship values. For example, D_f may represent the closeness between two friends in an online social network (family/close/normal/stranger) or the weight of an edge in a trust network[1] of the participants.

[1]trust.mindswap.org

Denote as $\mathcal{O} = \{0, 1\}$, the set of outcomes, corresponding to bad or good, of a transaction between peers. A peer provides a service with price u for its clients, where u lies within a range of minimal price u_* and maximal price u^*. In practice, a centralized system can define these values u_* for each category of services or products, or peers can learn these values by looking at the trading history of other peers in the system. Thus we name u the legitimate payoff (or gain) of a provider peer in a transaction if it behaves honestly, e.g., providing a service with high quality. The transaction in this case is considered as having a good outcome to the client. Note that if there is a small probability that a provider with honest behavior still yields a bad transaction outcome to the client, the inclusion of such a probability in our approach is straightforward. In the opposite case if the provider cheats (provides a bad service), the provider gains a further illegitimate amount υ, where $0 \leq \upsilon \leq \upsilon^* < \infty$ and the transaction outcome is considered as bad. The upper bound υ^* typically depends on the maximal price u^* of a service, and is only necessary in our theoretical analysis. In practice, rational peers need to know only υ for each transaction they one involved in. For example, υ approximates the service value as evaluated by the provider plus the shipping cost. Note that υ may be less than the service value u, e.g., the client still receives the service, which has worse quality than that of the original description.

We also define Ω as the set of all transactions in the system and $\Omega(x, y)$ the set of all transactions where peer x is the client and peer y is the provider. For convenience, let $S(x)$ be the set of peers having provided services to x. Let $r(x, y, tr) \in \mathcal{O} = \{0, 1\}$ be a binary rating from a peer x on another peer y on a transaction $tr \in \Omega(x, y)$.

2.3. Computational Trust Models as Dishonesty Detectors

A peer (the learning peer) may use a computational trust model to learn the rating behavior of another (the target). A trust model can be implemented with various statistical or heuristic methods and based on several information sources. These sources may include personal experience of the learning peer, the recommendations/ratings on the target and the target's intrinsic features, namely the frequencies of its ratings [Cornelli et al. 2002] or the benefit relationships of the rater and the rated peer [Ashri et al. 2005]. We propose using such a computational trust model as a dishonesty detector to evaluate the trustworthiness (reliability) of a rating on a provider. From this perspective, a trust mechanism can be abstracted as in Definition 2.1 (see Section 2.2 for the related notations).

Definition 2.1. A personalized reputation-based *computational trust model* used by a peer i to estimate the trustworthiness of a rating by another peer j is a 5-tuple $\mathcal{R} = \langle P_i, V_i, \mathcal{F}_j, A, D \rangle$ where:

— $P_i \subseteq P$ is a set of peers that i considered as relevant to the evaluation;
— $V_i \subseteq \{r(x, y, tr) \mid x, y \in P_i, tr \in \Omega(x, y)\}$ is the set of ratings related to peers in P_i;
— $\mathcal{F}_j$ is the properties of the target peer j, for instance, its location, its frequencies of posted ratings, the number of involved transactions, etc. and so on;
— A is an algorithm that operates on P_i, W, V_i, $\mathcal{F}_j$ and outputs a trust value T_{ij}, where W is the social relationships among peers. T_{ij} may take a binary, discrete, or real value, and is the estimated trustworthiness of j in posting the current rating; and
— D is the decision rule(s) with a binary outcome in $\{1, 0\}$, stating whether i should trust the rating by j given the personalized trustworthiness T_{ij}.

Definition 2.1 is an abstraction of several (if not most) popular heuristic or statistical trust evaluation algorithms in the literature, including those trust management approaches in peer-to-peer systems [Despotovic and Aberer 2006], social networks [Golbeck 2006], and other online communities [Dellarocas 2005b; Jøsang et al. 2007].

The appendix gives some examples using the formalism in Definition 2.1 to model some computational trust models proposed in Xiong and Liu [2004] and Vu and Aberer [2007]. Other trust models can be represented similarly. For instance, the global trust models like EigenTrust [Kamvar et al. 2003] and complaint-based [Aberer and Despotovic 2001] consider all peers in the networks as relevant to the computation, thus $P_i = P$, and consider the recommendation/ratings among each pair of them. Another simple yet important model is given in Example 2.2.

Example 2.2. The naive computational trust model $\mathcal{N} = \langle \emptyset, \emptyset, \emptyset, \mathcal{I}, D_{\mathcal{I}} \rangle$ uses an algorithm, $A = \mathcal{I}$, which always outputs $T_{ij} = 1$ and the set of decision rules, $D_{\mathcal{I}}$, which always considers a rating reliable. That is, the naive model $\mathcal{N}$ simply trusts all ratings by any peer.

Considering a computational trust model as a dishonesty detector, we define its statistical accuracy measures as similar to those of a conventional spam filter [Graham 2002] in Definition 2.3.

Definition 2.3. The *accuracy of a dishonesty detector* $\mathcal{R}$ in estimating the reliability of a rating is defined by its two *misclassification error rates*, the *false positive rate* α =Pr(a rating estimated as reliable by $\mathcal{R}$ | the rating is actually unreliable), and the *false negative rate* β = Pr(a rating estimated as unreliable by $\mathcal{R}$ | the rating is actually reliable).

The accuracy of a computational trust model implies its resilience to the possible malicious manipulation of the ratings. Although the actual values α, β, of a dishonesty detector may be unknown and change (possibly improve) over time, given the learning capability of the computational mechanism, we are mostly interested in the upper-bound ε of these errors. The values $\alpha, \beta, \varepsilon$, whichever are known, are common knowledge in the system.

2.4. Strategic Peer Selection Protocol

The following protocol is proposed for a rational peer, as a client, to select a provider among candidates (Definition 2.4). The concrete implementation of this selection protocol in a dynamic setting will be presented in Section 5.

Definition 2.4. A peer (a potential client) uses the following *peer selection protocol* $\mathcal{S}_k = \langle \mathcal{R}, k \rangle$ (where $k \geq 1$) to evaluate the eligibility of a provider for its transactions.

(1) The client gets the most recent binary rating r on the provider, considering the absence of a rating as the presence of a positive one.
(2) The binary reliability $\widehat{t}$ of r is evaluated with the computational trust model $\mathcal{R}$.
(3) If $\widehat{t} = 1 \wedge r = 0$ or $\widehat{t} = 0 \wedge r = 1$, the client publishes this information (a reliable detection of one cheating by the provider) to the shared space (a global black list).
(4) The provider is included for selection if in the shared space there are less than k published cheating detections on the provider, otherwise the provider is ignored.

Table I summarizes the most frequently used notations in the article. The parameter $k \geq 1$ represents the cautiousness of a peer in trusting the cheating detections published by the others. The evaluation of rating reliability with a computational trust model in step (2) is used to reduce influences of strategic manipulation of ratings by rational or malicious peers. The goal here is to eliminate as many malicious providers as possible when they cheat and motivate rationally opportunistic providers to cooperate. The use of this peer selection protocol with a global computational trust model mimics the behavior of a centralized reputation system in practice. The protocol $\mathcal{S}_k$ is tough for bad providers, including malicious and rationally opportunistic ones. Given

Table I. Commonly Used Notations

Notation	Definition
u_*, u^*	minimal and maximal price of the offered services, respectively
u	legitimate gain of a provider when cooperating, $u_* \leq u \leq u^*$
υ	additional gain by cheating of a provider in a transaction, $\upsilon > 0$
υ^*	maximal additional cheating gain of a provider, $0 \leq \upsilon^* < \infty$
$\mathcal{R}$	a computational trust model to estimate a rating's reliability (Definition 2.1)
$\mathcal{N}$	the naïve trust model in Example 2.2
α	Pr(rating estimated as reliable | rating is actually unreliable)
β	Pr(rating estimated as unreliable | rating is actually reliable)
ε	upper-bound of α and β, $0 \leq \varepsilon \leq 1$
k	# of posted cheating detections on a provider before it is globally ignored by other clients
$\mathcal{S}_k = \langle \mathcal{R}, k \rangle$	a peer selection protocol specified in Definition 2.4
Δ	# of remaining transactions in which a rational provider does not necessarily have incentives to cooperate, $\Delta > 0$

the high cost of identities, as previously assumed, it assures that a globally blacklisted provider has no further chance to provide services or gain revenues from the system. An extension of this analysis to the case of cheap identities is given in another work of ours [Vu and Aberer 2010].

We want to emphasize that this work is only specific in its assumption of the selection protocol. Though better protocols are feasible, the current one allows us to study the dependencies of the accuracy of learning and cooperation in the system, in a general way. The results obtained are generally applicable in case of any computational trust mechanism being used and in the presence of many realistic user behaviors; in fact, many existing reputation-based trust approaches are covered by this selection protocol.

2.5. Scope of Analysis

To make the problem tractable, we do not consider the following orthogonal issues. First, the incentives of peers to leave feedback after a transaction, are not dealt with directly in this work. Nevertheless, the absence of a rating after a transaction is seen as the presence of a positive rating, thus in the case of few ratings, appropriate decisions can still be made. Similarly, providing incentives to share data for the evaluation of a rating's reliability is an orthogonal issue. These incentives for sharing the learning results can be solved—a client can still do the learning by itself and our approach still holds if the evaluation of rating reliability is verifiable so that wrong evaluations can be detected. In centralized systems, any evaluation is performed by a single trusted entity and thus no verification is necessary. In decentralized systems, the learning in step (2) can be implemented to be verifiable by any peer by requiring the peers to also post data related to their evaluation, to the shared public space, or to provide such data on-demand for the querying peer(s), e.g., via an easy-to-validate form such as proof-carrying-code [Necula 1997]. Furthermore, it is possible to integrate existing incentive mechanisms, e.g., via side-payment [Miller et al. 2005] to motivate the sharing of feedback and evaluation results in the system.

Another potential problem is when many peers collude to badmouth a provider. Accidental blacklisting of a good peer is not very harmful to a client and can be reduced by increasing k and lowering the error of the rating evaluation with a more expensive and sophisticated trust model. A robust computational trust model $\mathcal{R}$ should consider the trustworthiness of both the rater and the rated provider to reduce the undesired impact of observation noise and badmouthing. The designing of such a robust and accurate computational trust mechanism, however, is orthogonal to the current work.

3. USING COMPUTATIONAL TRUST MODELS TO FOSTER COOPERATION

3.1. Quantifying a Computational Trust Model's Accuracy

The misclassification errors of a given computational model depend on several factors, the most important being the design of the model.

Proposition 3.1 presents a preliminary analysis on the accuracy measures of a computational trust model used by a rational peer and can be seen as one guideline to implement an appropriate dishonesty detector in our approach. The proof can be found in the electronic appendix accessible in the ACM Digital Library.

PROPOSITION 3.1. *Given a computational trust model* $\mathcal{R} = \langle P_i, V_i, \mathcal{F}_j, A, D \rangle$ *(c.f. Def. 2.1) publicly known to every peer. Suppose that raters are rational and want to maximize the misclassification errors of the detection. If* D *yields a deterministic and publicly known outcome at each evaluation step, in equilibrium* $\alpha = \beta = 0.5$*. The use of the model* $\mathcal{R}$ *as a dishonesty detector thus yields no better result than random guessing.*

We are concerned with those computational models that have the upper-bound of misclassification errors $\varepsilon < 0.5$. Although Proposition 3.1 shows cases where such bounds cannot be achieved, practically accurate learning with α, β upper-bounded by some $\varepsilon < 0.5$ is feasible. We claim that the details of algorithm A, e.g., the rating history to be used for the estimation, or the decision rule D can be kept as private information of the learning peer before the evaluation (after which the related data can be published to enable verification). Misclassification errors of the associated computational trust model as a dishonesty detector can be measured by empirical experiments. This question has already been extensively studied in previous work, namely Xiong and Liu [2004] and Kamvar et al. [2003], most of which have shown that low α, β can be achieved under various rating manipulation attacks. In Section 6.4 we will also present measurements on the errors α, β of popular computational trust models and confirm that achieving an accuracy bound $\varepsilon < 0.5$ is possible. The naive computational trust model $\mathcal{N}$, which trusts any rating and considers the absence of a rating as a positive one, has the misclassification errors $\alpha = \alpha_0 = 1$ and $\beta = \beta_0 = 0$ (see Example A.3 in the appendix).

Another example of an accurate yet expensive method to estimate the reliability of ratings is to perform full monitoring on performance of the provider to learn its real past behavior. Such a monitoring can be implemented in many ways. In an e-commerce system, it is possibly done via legal investigations of suspicious transactions. In a market of e-services, one can deploy monitoring agents to periodically probe and test the service being offered by a provider to estimate the real offered quality level offered by that provider during a specific period.

3.2. Using Trust Models for Cooperation Enforcement

Given a bound ε for α, β of the computational trust model(s) being used by clients in the system, Theorem 3.2 shows the relation between the error bound ε of a computational trust model and its effectiveness in enforcing cooperation of a provider during its lifetime. Again the proof can be found in the electronic appendix.

THEOREM 3.2. *Given the peer selection protocol* $\mathcal{S}_k = \langle \mathcal{R}, k \rangle$*, where the computational trust model* $\mathcal{R}$ *has the misclassification errors* α, β *upper-bounded by* $\varepsilon < 0.5$*. Suppose that identities are difficult or very costly to obtain for any participant. If we have in addition* $\varepsilon < \varepsilon_{max}(k) = 1/(1 + \sqrt[k]{1 + v^*/u_*})$*, then with* Δ_v *defined as:*

$$\Delta_v = max\{1, \lceil \frac{\ln [1 - \frac{v\varepsilon^k}{u_*((1-\varepsilon)^k - \varepsilon^k)}]}{\ln (1 - \varepsilon^k)} \rceil\}, \tag{1}$$

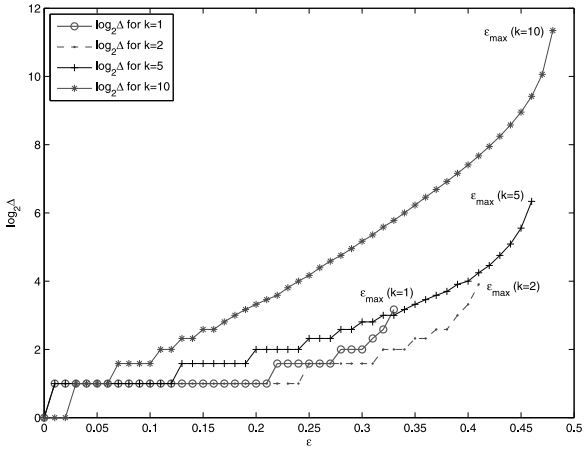


Fig. 1. The relation between ε and Δ for different values of k and $u_* = \upsilon^*$. A rational provider finds it most beneficial to cooperate during all but its last Δ transactions.

(1) *it is optimal for a rational provider to cooperate in the current transaction, whose additional cheating gain is υ, if the provider stays for more than Δ_υ further transactions;*
(2) *a rational provider considers cooperation as its best response strategy in all but its last $\Delta = \Delta_{\upsilon^*}$ transactions.*[2]
(3) *Let N_h be the number of transactions in which an honest provider can participate till mistakenly blacklisted, and let N_c be the number of bad transactions a malicious provider can defect until globally eliminated from the system, respectively, then $E[N_h] > 1/\varepsilon^k$ and $E[N_c] < 1/(1-\varepsilon)^k$.*

These results hold even in the presence of strategic manipulation of ratings by providers.

Theorem 3.2 also holds if the clients use different computational trust models with different inputs and personalized settings to evaluate the trustworthiness of the last rater, and the probability of detecting a bad rater is different for each peer. The analysis can also be extended to include the probability that an honest provider appears as cheating, e.g., a good seller rated negatively for an article lost during shipping, or the probability a cheating provider still satisfies the client, e.g., a seller sends a low-quality article yet still pleases the client.

Naturally, $\Delta \to \infty$ with very high values of the cheating gain υ^*, such as when expensive articles are sold. Enforcing the cooperation of a rational provider in such expensive transactions is impossible, which is intuitive. In other cases, a relation between Δ and the error upper-bound ε can be drawn, as in Figure 1 with $\upsilon^* = u_*$, i.e., peers sell and buy items of comparable prices. We have the following observations. First, the required upper-bound $\varepsilon_{max}(k)$ reaches 0.5 with larger k values, yet the incentive of cooperation decreases rapidly (Δ becomes larger). Even so, for rational long-term providers who stay in the system infinitely, the number of last transactions Δ plays no role and thus any trust model with a reasonably good accuracy $\varepsilon < \varepsilon_{max}(k)$ can be used as an effective sanctioning tool to motivate providers' cooperation (Corollary 3.3). Given an approximate value of ε, one should select the threshold k appropriately such that $\varepsilon < \varepsilon_{max}(k)$. For a given $\varepsilon < \varepsilon_{max}(k)$, smaller threshold k is

[2]With small ε^k, $\Delta_{\upsilon^*} = max\{1, \lceil \upsilon^*/(u_*((1-\varepsilon)^k - \varepsilon^k))\rceil\}$, as in a previous work [Vu and Aberer 2008].

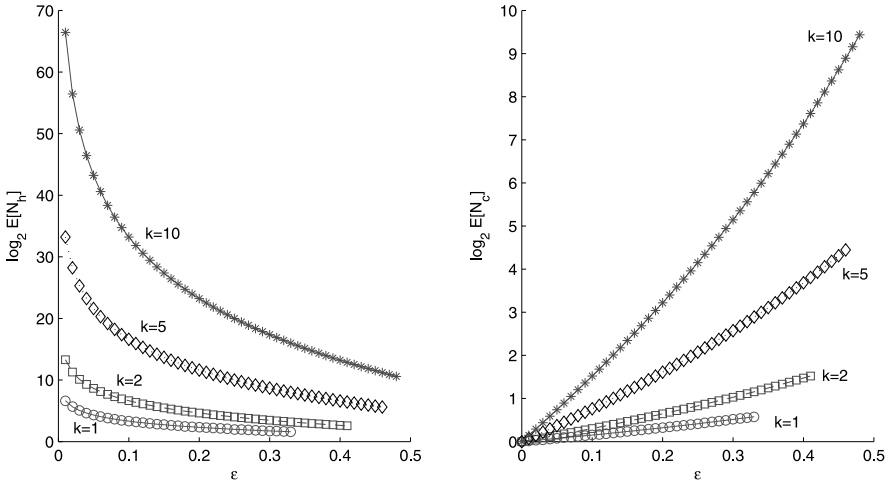


Fig. 2. Relation between ε and a lower-bound of $E[N_h]$ (left), and an upper-bound of $E[N_c]$ (right).

preferred. In certain application contexts where the providers only participate in a limited number of transactions, or in cases with high k values, very high levels of accuracy ($\varepsilon < 0.05$) are required to reduce the parameter Δ, i.e., to ensure cooperation of providers in most transactions.

Note that we do not offer any conclusions on the incentives of cooperation of a rational provider when selling its last Δ services. In fact the provider may find it beneficial to even cooperate in these last transactions, if the temporary cheating gain is small compared to the probability of being blacklisted. This was shown in our experiments where fully cooperative providers who never cheat even in their last transactions have higher utilities than the strategic providers who may defect when providing some of their last Δ services.

COROLLARY 3.3. *It is possible to use any computational trust model with misclassification errors upper-bounded by some $\varepsilon < \varepsilon_{max} < 0.5$ to effectively enforce cooperation of rational providers who participate infinitely or in a very large number of transactions, even in the presence of strategic rating manipulation by participants.*

The possibility of correctly eliminating the malicious providers and wrongly blacklisting the honest ones is given in Figure 2. The system behaves much better with lower ε, as intuitively expected. The higher the accuracy of the computational trust model being used (lower ε), the higher the expected number of transactions an honest provider may stay in the system (higher $E[N_h]$), and the lower the survival chance of the malicious providers (lower $E[N_c]$). Also, higher thresholds k, reduce the possibilities of wrongly blacklisting honest providers yet also increase the incentives of malicious behaviors. Hence, given a known ε, it is recommended to choose the value of k appropriately, depending on the prior information on the environment vulnerability. In vulnerable environments, smaller k values are better used by clients to quickly eliminate bad providers at the cost of ignoring good providers. In less vulnerable systems higher k is recommended. Note that the trend given in Figure 2 is for the worst case scenario where an honest provider is repeatedly badmouthed by other users, and a malicious provider has enough resources for disguising her cheating activities by consecutively posting many positive ratings to the system.

Our analysis holds if there exists an effective identity management scheme to ensure that blacklisted providers have to pay a high cost to enter the system again, and hence identities are not cheap. This is realistic in many practical applications, e.g., in our running example of trading on a social network such as Facebook, users have to spend time and money to build strong relationships with other users, so it is nonoptimal for them to simply cheat, discard their long-time investment in their current identities and start all over again. Our analysis also provides a starting point to design a mechanism to ensure full cooperation of rational providers, either by (1) explicitly imposing a sufficiently high entrance cost $c \geq \Delta v^*$ for newcomers, or (2) by giving advantages to a provider with a longer interaction history, e.g., by matching the provider to more clients and increasing the value of its identity to demotivate it from cheating and escape. An extension regarding (2) is given in another work [Vu and Aberer 2010].

3.3. Using the Naive Computational Trust Model to Foster Cooperation

Corollary 3.4 shows the relation between the capability of the naively optimistic trust model $\mathcal{N}$ (Example 2.2) in enforcing cooperation and the client's truthfully reporting probability h, assuming no strategic rating manipulation. The selection protocol $\langle \mathcal{N}, 1 \rangle$ is actually equivalent to the reputation system with only the last rating studied by Dellarocas [2005a].

COROLLARY 3.4. *Suppose that* $1 \geq h > h_{min} = (1 + v^*/u_*)/(1 + 2v^*/u_*)$ *is the average probability that a client leaves an honest rating after a transaction. Without strategic rating manipulation, the peer selection protocol* $\langle \mathcal{N}, 1 \rangle$ *makes it optimal for a rational provider to cooperate in all transactions but its last* $\Delta = max\{1, \lceil \frac{\ln [1-(1-h)/(2h-1)]}{\ln h} \rceil\}$ *ones. Such a selection mechanism also gives direct incentives for long-term rational clients to leave truthful ratings after their transactions.*

The assumption of no strategic rating manipulation in Corollary 3.4 implies that users submit a rating based only their erroneous evaluation of a provider's behavior. In the presence of strategic manipulation of the reports, a client must use a sophisticated trust-learning algorithm to evaluate the reliability of the rater, as presented in the Section 3.2. Otherwise a selection protocol using only the latest rating can be easily attacked. After cheating in a transaction a provider may collude with another client to stuff a positive rating on a new fake transaction to hide its malicious behavior. Figure 3 shows the relation between the overall probability h a peer is an honest reporting peer vs. the number of Δ last transactions for which rational providers do not find incentives to cooperate, for different v^* and u_*. The cases where $v^*/u_* < 1$ correspond to when a provider sells a service with a lower quality than it promises. Let us consider the previous trading scenario where peers sell and buy services of similar prices ($u_* \simeq v^*$). If the provider is a long-term player, and the honest rating probability $h > h_{min} \simeq .8$ (a highly noisy environment), then it is optimal for a provider to behave honestly for most of its transactions, except the last two.

4. COST-EFFICIENT REPUTATION MANAGEMENT

While a computational trust model with a better accuracy is generally preferable (see Theorem 3.2), it usually comes with higher cost. The implementation cost of a computational trust model $\langle P_i, V_i, \mathcal{F}_j, A, D \rangle$ (Def. 2.1) may consist of several components.

— *The communication cost* τ_c to explore the peers P_i and retrieve the relevant ratings V_i. Associated with this cost is the storage cost τ_s to store and to maintain rating information and historical performance of potential partners/raters.

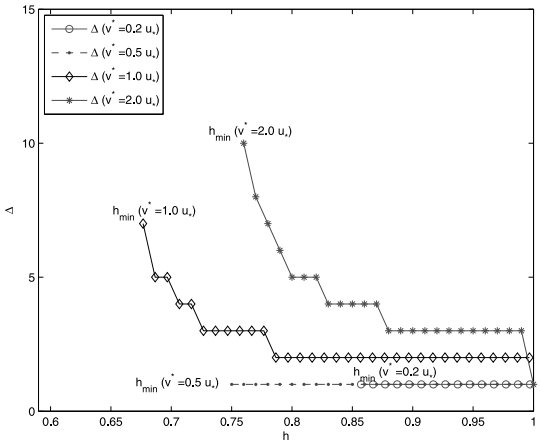


Fig. 3. The relation between the truthful reporting rate h and the number of last transactions during which a rational provider has no incentive to cooperate in case its clients use the selection protocol $\langle \mathcal{N}, 1 \rangle$.

— *The computational cost* τ_e of the algorithm A and decision-making algorithm $\mathcal{D}$ to aggregate, analyze ratings and make appropriate decisions.
— *The monetary cost* τ_m. Reliable ratings can be considered as sellable goods, thus buying them incurs certain costs. τ_m may also include the payment for the participants to elicit their truthful reporting in some reputation mechanisms, such as Miller et al. [2005].
— *The hidden/opportunity cost* τ_o. Time-consuming trust learning algorithms may lead to late decisions and be more costly, especially in competitive scenarios.

A detailed study of the cost model of each computational model is system and application-dependent. For example, the estimation of the significant opportunity cost τ_o during the lifetime of a peer is nontrivial. Such a study of all cost types of existing algorithms is out of the scope of this work. We only focus on the communication cost τ_c, and the related cost measurements for typical computational trust models, which are be given in Section 6.

Consider the case where a peer can select between a computational trust model $\mathcal{R}_1$ or another model $\mathcal{R}_2$, to evaluate the most recent rating on a provider. Suppose that $\mathcal{R}_1$ is an accurate algorithm with misclassification errors α, β upper-bounded by a small $\varepsilon < \varepsilon_{max}(k = 1)$, and with an expected cost $\mathcal{C}_1$. An example $\mathcal{R}_1$ is to buy information from some third-party monitoring agents to get an estimate of the provider's behavior on the last transaction. Let $\mathcal{R}_2$ be the naive computational trust model $\mathcal{N}$ that trusts any rating, which has the misclassification errors $\alpha_2 = 1, \beta_2 = 0$ and a negligible cost $\mathcal{C}_2 \ll \mathcal{C}_1$. Since $\alpha, \beta < \varepsilon < \varepsilon_{max}(k = 1)$, the expensive model $\mathcal{R}_1$, can be used to motivate the cooperation of the providers in most of their transactions (Theorem 3.2). Using the naive model $\mathcal{N}$ is cheaper and preferable, yet it is impossible to use only this naive model to enforce cooperation since the provider may strategically manipulate its ratings by colluding with the raters. Theorem 4.1 proposes a way to optimize the cost of using the expensive computational model $\mathcal{R}_1$, while still ensuring cooperation by rational providers (the proof is in the electronic appendix).

THEOREM 4.1. *Consider the selection protocol* $\mathcal{S}_1 = \langle \mathcal{R}, 1 \rangle$*, in which the dishonesty detector* $\mathcal{R}$ *is implemented by using the trust model* $\mathcal{R}_1$ *with probability* c *and the naive model* $\mathcal{N}$ *with probability* $1 - c$*. Suppose that* ε *is small and the incentive for badmouthing is negligible, e.g., providers sell different services with little competition.*

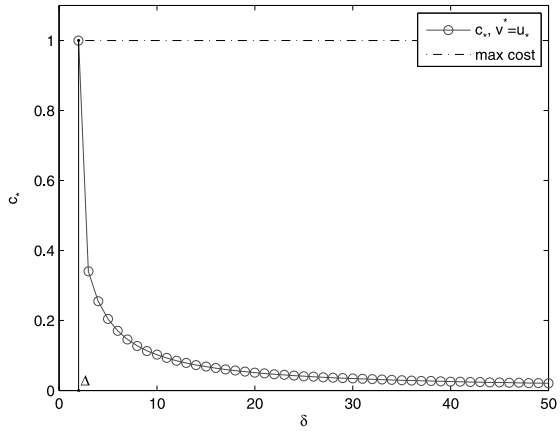


Fig. 4. The minimal necessary probability c_* to use the expensive algorithm $\mathcal{R}_1$ depending on the number of remaining transactions δ of a rational provider for $\varepsilon = 0.01$, $u_* = v^*$.

The provider finds it optimal to cooperate in all but its last Δ *transactions, where* $\Delta = max\{1, \lceil \frac{v^*}{u_*(1-2\varepsilon)} \rceil\}$*, if* $c \geq c_* = \frac{v^*}{\delta u_*(1-2\varepsilon)}$*, where* $\delta \geq \Delta$ *is the number of remaining transactions of the rational provider. This result holds in the presence of strategic manipulations of ratings.*

Figure 4 shows the probability c_* that a client must use the expensive trust model $\mathcal{R}_1$ given the estimated remaining transactions δ of the provider, for $u_* = v^*$, $\varepsilon = 0.01$. If most providers stay in the system infinitely or long enough, the mixture of two computational trust models helps reduce the total implementation cost significantly. Hence, given the rationality of participants, the accurate computational trust algorithm $\mathcal{R}_1$ mostly plays the role of a sanctioning tool rather than a tool for learning the trustworthiness of the participants.

5. USING REPUTATION INFORMATION IN DYNAMIC SCENARIOS

The preceding analysis shows that if rational peers stay in the system infinitely, cooperation is ensured relatively easily. Based on this result, we present an implementation of the peer selection protocol such that the rational providers are motivated to cooperate in most of their transactions, given that peers may arrive and leave at different times. Note that we still restrict the problem to the case where the cost of establishing identity is high, and each peer is motivated to use one identity for all transactions.

We approximate the algorithm for a rational (strategic) client to select a provider before each transaction as in Algorithm 5. The algorithm is built on the observation that the distribution of the number of lifetime transactions of providers can be learned from the system history and available as common knowledge. Rational peers use this information to adapt their behavior strategically to maximize their long-term benefits.

Algorithm 5 follows the peer selection protocol $\mathcal{S}_k = \langle \mathcal{R}, k \rangle$ in Definition 2.4, where the involved quantities are computed trivially. For example, in an e-trading system, the minimal legitimate gain u_* of a provider (a seller) is the minimal value of an article accepted for trading in the system. The gain v is the value of the current item plus the shipping cost announced by the seller. Δ_{min} is the minimal number of remaining transactions that a rational provider finds incentives to cooperate for the worst level of α, β (claim (1) of Theorem 3.2). As the distribution of the number of transactions by providers can be learned from trading history of all providers, it is possible for the

Algorithm 1 selectProvider(gains u, υ, providers S, alg $\mathcal{R}$, threshold k)

1: *Eligibles* = $\varnothing$;
2: Retrieve the global blacklist L;
3: **for** each $s \in S \setminus L$ **do**
4: Get worst case values ε of errors α, β of algorithm $\mathcal{R}$;
5: Estimate benefit u_* and current cheating gain υ of s;
6: Compute Δ_{min} as in Eq (1) with υ;
7: $p[s]$ = Pr[provider s stays at least Δ_{min} further transactions];
8: Get binary rating r_i by peer i on the latest transaction of s;
9: Run $\mathcal{R}$ to evaluate the reliability (binary trust) t_i of r_i.
10: **if** $r_i == t_i$ **then**
11: *Eligibles* = *Eligibles* $\cup$ $\{s, p[s]\}$;
12: **else**
13: Post the cheating detection $r_i \neq t_i$ on the shared space;
14: Put s in the blacklist L if there are at least k such negative results;
15: **end if**
16: **end for**
17: Select the provider s from *Eligibles* with probability $p(s)/\sum_s p(s)$;

Algorithm 2 bestServiceStrategy(alg $\mathcal{R}$, threshold k): servingStrategy

1: Get worst case values ε of errors α, β of algorithm $\mathcal{R}$;
2: Estimate the minimal own benefit u_* and illegitime gain υ in the current transaction;
3: Compute Δ_* as in Eq (1) with υ;
4: Estimate its own remaining number of transactions δ;
5: **if** $\delta \geq \Delta_*$ **then**
6: Return cooperative;
7: **else**
8: Return cheating;
9: **end if**

client to estimate the probability $p[s]$ that a provider stays in Δ_{min} further transactions. This estimation is personalized to each client and dependent on his own belief in the continuation of the game of the provider with future clients. The probability $p[s]$ can be seen approximately as the probability that this strategic provider cooperates in the current transaction and thus is used as a selection criteria (line 17). It is noteworthy that even if $p[s]$ is known to the provider, the recursive end-game situation still does not occur. As long as the provider uses a single identity for selling services, the strategy of Theorem 3.2 is still the best for them.

According to Theorem 3.2, the best strategies of rational providers are implemented as in Algorithm 2. That is, a strategic provider will cooperate in all transactions except its last Δ_* ones. Similar to its clients, the provider estimates its Δ_* parameter based on the global knowledge of the misclassification errors α, β of the learning algorithm being used, its current temporal gain υ, and the minimal legitimate gain u_* at each step.

Given Algorithms 5 and 2, the number of good transactions depends on how accurately the clients can estimate the number of remaining transactions of a provider in order to select the right one still having incentives to cooperate. More concretely, a strategic provider is motivated to cooperate if $\delta \geq \Delta_*$.

The algorithm for a strategic client to use a mix of two trust models $\mathcal{R}$ and $\mathcal{N}$ to select a provider (Theorem 4.1) is implemented as follows. A client estimates the

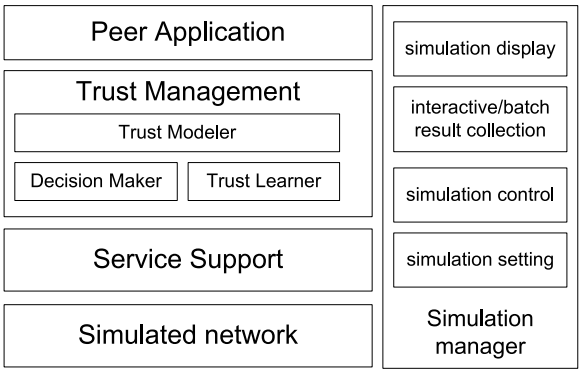


Fig. 5. Architecture of the trust prototyping and simulation framework.

number of remaining transactions δ_{client} of a provider from the expected number of transactions of all providers in the system. Next, it computes the minimal probability $c_{client} = \frac{v}{\delta_{client} u_* (1-2\varepsilon)}$ it needs to use $\mathcal{R}$ in the current transaction. On the other hand, a strategic provider also estimates the values c_{client}, δ_{client}, and computes the minimal probability $c_{provider} = \frac{v}{\delta u_* (1-2\varepsilon)}$ for its actual remaining number of transactions δ. The provider cooperates only if it stays more than $\delta > \Delta_*$ and $c_{client} \geq c_{provider}$.

Since it is difficult to analytically measure the system's cooperation level with various join and leave processes of peers, this measurement will be done later via a simulation whose input parameters are obtained from real case studies (Section 6).

6. EXPERIMENTAL RESULTS

6.1. Simulation Framework

We have developed a generic trust-prototyping and simulation framework and used it for the experiments in this work. This tool enables the rapid development and testing of many computational trust models under a variety of settings. Particularly, new algorithms for peers and system performance metrics can be defined and integrated into the framework without major effort.

Figure 5 shows the overall architecture of the framework, which is composed of the following modular components. The PeerApplication layer is an abstraction of the P2P application to be simulated, which is defined as an interaction scenario where peers provide or consume resources of different prices and quality. This abstraction makes it possible to tweak our system to simulate a range of similar applications, e.g., a customer-to-customer trading scenario, or a market of services. The TrustManagement layer serves as a trust management subsystem to help a peer to model and learn the trustworthiness of another. APIs and basic building blocks are given for users to develop and plug in many computational trust learning models and decision-making algorithms into the system. The ServiceSupport layer encapsulates the distributed information storage, retrieval, and routing mechanisms being used. This layer provides the upper layers with capabilities of sending requests and receiving responses among the peers, as well as enabling the retrieval of information on the services and their ratings for the trust computations. As a result, this layer facilitates the integration and testing of a given trust management approach on top of different distributed information management mechanisms. The SimulatedNetwork layer represents the underlying communication network, which can be implemented as a centralized or decentralized system depending on the simulated system. As we only emphasized

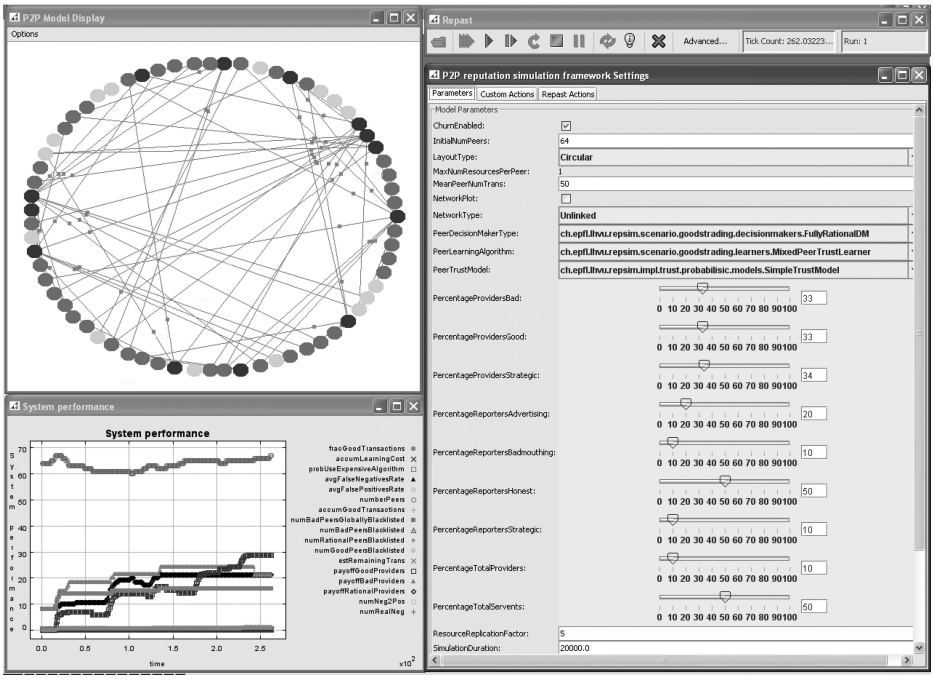


Fig. 6. Screenshots of the trust-prototyping and simulation framework in the interactive mode. Users can configure several simulation parameters (the right panel) such as the percentages of different peer types and the computational trust models to be used. The trust relationships among peers are shown in the top-left panel, where honest participants have many trust relationships. During simulation, the behavior of peers can also be inspected and changed interactively or programmatically. The system performance over time is shown in the bottom-left panel. Most simulation settings such as the distribution of peer types with various behaviors and the definitions of new system performance metrics can be done programmatically via the provided APIs.

the study of the social and strategic behavior of peers in the application layer, we only provided a basic implementation of this layer. The layer was implemented as a network of nodes with the latencies among them following the King latency data set [Gummadi et al. 2002]. In fact, our system can be configured to use other message-passing protocols or underlying communication networks with any latency models, as needed. The SimulationManager takes care of all back-end supports for the simulation, e.g., the measurement, inspection, and collection of system performance data in both interactive and batch modes. We used RePastJ [North et al. 2006] as the base to develop this component, as this library provides several useful tools for setting up the environment, dynamically controlling simulation parameters, result visualization and analysis, and so on. Again, the modular design and well-specified APIs of the SimulationManager layer makes it possible to use the system with other simulation libraries. In fact, it is possible to develop a decentralized implementation of this layer to support larger-scale simulations or even to emulate an application scenario on a real network, given participation of real users. Figure 6 shows some screenshots of our prototyping and simulation framework in action. Further details can be found online.[3]

[3]http://lsirpeople.epfl.ch/lhvu/download/repsim/

6.2. Simulation Goals and Settings

Since it is nontrivial to implement full rationality, rational peers were approximated as strategic: they used all available information to make the best decisions to maximize their expected lifetime utilities. We measured the cooperation level in the system given that the rational participants had knowledge of the theoretical results proved in the previous analysis. Specifically, the goal of our experiments was to investigate, (1) whether reasonably accurate computational trust models helped to enforce cooperation in such approximately rational environments even with observation noise, and (2) whether expensive trust models could be used less frequently to save cost without considerably affecting cooperation.

We used the trust simulation framework to implement an example P2P trading application. Peers were modeled as sellers or buyers of many articles at a similar price and exhibited different behaviors. The cooperation level and accumulated utilities of different types of peers were then measured and analyzed under various simulation settings. First, peers could leave and join dynamically, thus having different numbers of transactions during their lifetime. Second, buyers used different computational trust models with personalized inputs to evaluate the trustworthiness of the others. Third, peers exhibited several behaviors, both irrationally malicious and/or strategic—details to be presented in coming sections. The dynamic joins and leaves of peers in the system were simulated according to statistics from real-life case studies on many live peer-to-peer systems [Stutzbach and Rejaie 2006]. We set the interarrival time of peers to follow a Weibull distribution with shape parameter 0.53, and the up-time of peers to follow a Weibull distribution with shape parameter 0.34 [Stutzbach and Rejaie 2006]. Since we did not have statistics on the number of transactions a peer performs in a real peer-to-peer trading scenario, such a distribution was approximated as follows. We computed the number of transactions a seller participated in from its up-time by using a scaling factor K. K was set such that those peers with up-time approximating the mean of the overall uptime distribution would participate in μ_{trans} transactions, where μ_{trans} was a parameter of the simulation. The rationale behind this choice was that the number of transactions a peer was involved in was assumed to be proportional to its participating time in the system. Experiments with other distributions of the number of peer transactions are subject to future work.

6.3. Implementation of Peer Behaviors

A buyer first searched for the available articles and then selected one according to Algorithm 5 and its variant, as introduced in Section 5. A *good seller* shipped the article after receiving the payment with a high probability ($\gamma^+ = 0.99$) of satisfying the buyer and getting a good rating. A *bad seller* either did not ship the article or only shipped a low quality item to the buyer, resulting in a very low probability ($\gamma^- \approx 0.05$) of meeting buyer's anticipation and thus was likely to receive a bad rating afterwards. Other sellers were *strategic* and used available information to find the best strategy to follow, e.g., whether it should ship the item or not, so as to maximize its long-term utility, following the strategy described in Algorithm 2 (Section 5). Note that it is also possible to configure the simulation such that the observation noise γ^+, γ^- varies depending on other factors, e.g., subject to the buyer's viewpoint given the price of an item. However, this extension is out of the scope of the article and subject to future work.

Regarding rating behaviors, a peer exhibited one of the following behaviors: *honest* (always reported correctly about what it had observed), *badmouthing* (always reported negatively), *advertising* (always reported positively), *ignoring* (did not report), and *strategic*. Strategic raters might provide correct, incorrect ratings, or not leave any

Table II. Experimental Settings of the Representative Simulation Scenarios

Sellers consisted of: s% strategic, g% good, and b% bad, and raters are of: h% honest, sr% strategic, a% advertising, b% badmouthing, and the rest leaving no ratings after a transaction. We set $a > b$ since in most case studies of current reputation systems, the majority of the feedback is positive, thus it is likely that advertising behaviors are more popular. By default $k = 1$ if unspecified otherwise.

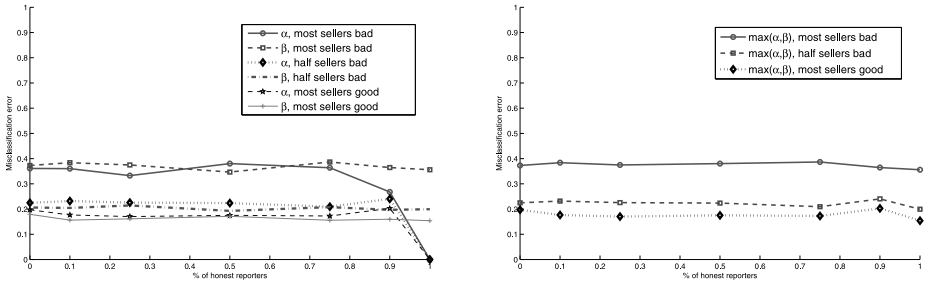
Scenario	s	g	b	h	sr	a	b	**Result**
C_1. No strategic sellers, most sellers bad	0	15	85	0 to 100	$\frac{100-h}{5}$	$\frac{2(100-h)}{5}$	$\frac{100-h}{5}$	Figure 7
C_2. No strategic sellers, half sellers good	0	50	50	0 to 100	$\frac{100-h}{5}$	$\frac{2(100-h)}{5}$	$\frac{100-h}{5}$	Figure 7
C_3. No strategic sellers, most sellers good	0	85	15	0 to 100	$\frac{100-h}{5}$	$\frac{2(100-h)}{5}$	$\frac{100-h}{5}$	Figure 7
C_4. Few sellers strategic, most bad	10	5	85	0 to 100	$\frac{100-h}{5}$	$\frac{2(100-h)}{5}$	$\frac{100-h}{5}$	Figure 8
C_4^k. Similar to C_4 with the threshold $k = 1, 2, 5$	10	5	85	0 to 100	$\frac{100-h}{5}$	$\frac{2(100-h)}{5}$	$\frac{100-h}{5}$	Figure 9
C_5. Most sellers strategic	85	5	10	0 to 100	$\frac{100-h}{5}$	$\frac{2(100-h)}{5}$	$\frac{100-h}{5}$	Figure 8
C_6. Approximately equal seller types	33	34	33	0 to 100	$\frac{100-h}{5}$	$\frac{2(100-h)}{5}$	$\frac{100-h}{5}$	Figure 8
C_6^k. Same as C_5, with the threshold $k = 1, 2, 5$	33	34	33	0 to 100	$\frac{100-h}{5}$	$\frac{2(100-h)}{5}$	$\frac{100-h}{5}$	Figure 9
C_7. All sellers strategic	100	0	0	0 to 100	$\frac{99-h}{3}$	$\frac{(99-h)}{3}$	1	Figure 10

rating, depending on the situation. To reduce the complexity of the simulation, in this work the rating behavior of strategic peers was limited to the following *safe* strategy. When A was asked by B how A estimated the trustworthiness of another peer X, the reporting strategy of A was determined based on the relationships $A-B$ and $A-X$ as follows. If the querying peer B was unknown to A, and the target peer X was a trusted peer, a blacklisted, or an unknown peer to A, A would respectively report positively, negatively, or honestly. If B was trusted by A, A always reported honestly. Requests from blacklisted peers were ignored by A. This reporting strategy was chosen for two reasons. First, the cost of reporting after a transaction in our current application was negligible. Second, this strategy helped peers to rapidly build up and extend its trust relationships with many other peers in the system, thus intuitively most beneficial to them.

Experimental settings and corresponding results are summarized in Table II. The trading system was simulated with dynamical leaves and joins of peers, starting with $n = 256$ peers. In the stationary regime, the number of peers approximately doubled this initial number. The simulator was able to run with up to 1024 initial peers, and the results were similar. Each simulation was run until the stationary regime and the measures of each metric were their medians over at least 35 different runs (to avoid outliers).

6.4. Accuracy of Example Computational Trust Models

Three computational trust models were used in the simulation to evaluate the reliability of the most recent rating on a seller. The first computational trust model $\mathcal{L}$ was the PeerTrust PSM/DTC algorithm proposed by Xiong and Liu [2004]. A peer i estimated the trustworthiness of another rater j based on the similarity between i's and j's ratings on some other sellers that both i and j had contacted. The second model $\mathcal{X}$ used the maximum likelihood estimation-based learning algorithm [Despotovic and Aberer 2004; Vu and Aberer 2007] with a probabilistic decision rule. With this model, a peer i estimated the probability a rater j was trustworthy to maximize the likelihood of getting the current ratings from i and j on those sellers with whom both i, j had experience. We also implemented a dishonesty detector $\mathcal{A}$ with a very good misclassification error bound $\varepsilon = 0.01$, and with a high cost of each time being used. In practice, a peer implements such an accurate detector by asking for information from the third party monitoring or consulting agents before becoming involved in important transactions. The dishonesty detector $\mathcal{A}$ was similar to a global computational trust model and used to verify the relation between the learning accuracy and the cooperation level in the



(a) The errors α, β of PeerTrust model $\mathcal{L}$ (b) The approximate error ε of model $\mathcal{L}$

Fig. 7. Accuracy of the trust model $\mathcal{L}$ in three scenarios C_1, C_2, C_3 of Table II, where $\mu_{trans} = 50$.

system, where peers used the same algorithm to evaluate the reliability of raters. The two algorithms $\mathcal{L}$ and $\mathcal{X}$ were used to test the efficiency of the peer-selection protocol in a more relaxed environment, where peers used different localized algorithms with personalized inputs and settings to estimate rating behaviors. For the sake of readability, in the following experiments only the results for the computational trust model $\mathcal{L}$ are shown, unless specified otherwise. The results for the model $\mathcal{X}$ were similar to those of $\mathcal{L}$, and the results of $\mathcal{A}$ were even better. The use of other global trust models like a complaint-based algorithm [Aberer and Despotovic 2001] or EigenTrust [Kamvar et al. 2003] instead of algorithm $\mathcal{A}$ is subject to future work.

As a base for other experiments, first the overall misclassification errors α, β of the computational trust models $\mathcal{L}$ and $\mathcal{X}$ were estimated for a variety of scenarios with different fractions of the honestly reporting users h in the system. The most representative scenarios C_1, C_2, and C_3 for this experimentation are given in Table II. These scenarios were designed based on the observation that it was unnecessary to measure precisely the accuracy of a learning algorithm, but only the overall trend and worst case misclassification errors α, β depending on the percentage of honest reporting peers h in the system. Furthermore, strategic selling behaviors could be excluded when estimating these misclassification errors since the learning used only historical data, and the outcome of strategic selling behaviors could be assumed to follow one of the overall trends of the three extreme cases C_1, C_2, and C_3. These statistics were measured for three cases with different fractions of good and bad sellers in the systems (scenarios C_1, C_2, and C_3 in Table II). Note that the accuracy of these algorithms has already been extensively studied in related work [Despotovic and Aberer 2004; Xiong and Liu 2004], most of which having been shown to have low α, β under various attack scenarios. In this article, the only difference was that such computational trust models are used to evaluate the reliability of the last rating on a seller, instead of estimating the trustworthiness of the seller. This reliability was determined based on the reputation information of the user giving that very rating. Hence, the goal of the simulation here was to verify that misclassification errors of these trust models, when used in this new context, were still sufficiently low, so that the theoretical analysis of the relation between the error bound and cooperation, would be applicable. That said, we needed to verify if an approximate bound ε of the misclassification error rates α, β of the computational trust model being used satisfies the requirements to use Theorem 3.2, i.e., $\varepsilon < \varepsilon_{max}$ for a chosen threshold k. The obtained error bound ε would then be used as common knowledge in later experiments with various combinations of strategic, bad, and good sellers.

Figure 7 shows the estimated maximal values of α, β of the learning algorithm $\mathcal{L}$ based on the PeerTrust PSM/DTC approach [Xiong and Liu 2004], where $\mu_{trans} = 50$,

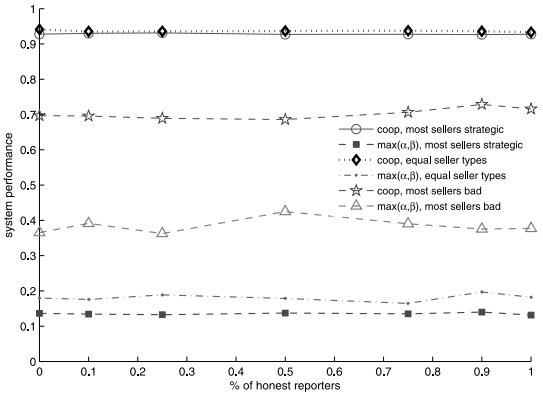
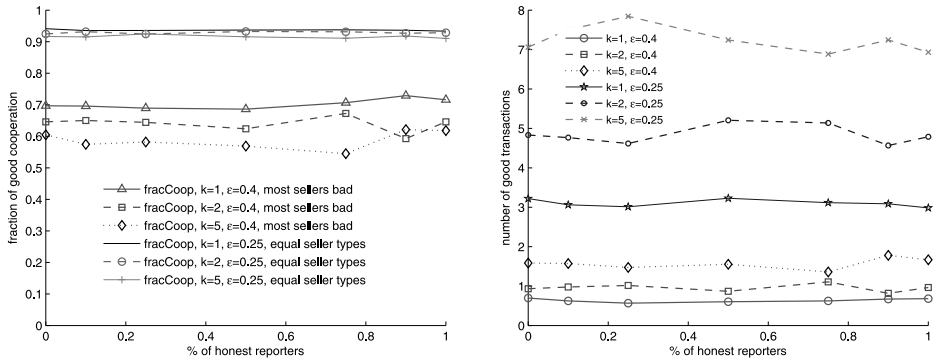


Fig. 8. Relation between the fraction of good transactions and the accuracy of the computational trust model $\mathcal{L}$ in the presence of honest, malicious, and strategic peers. In the cases of most strategic sellers (C_5) and with equal seller types (C_6), cooperation is very high thanks to the low misclassification errors. Cooperation is much lower with a lower reliability in the evaluation of the raters' behaviors (case C_4).

and in three extreme cases C_1, C_2, C_3 of Table II. Generally, the false positive error rate α of $\mathcal{L}$ was much lower where most peers staying in the system longer ($\mu_{trans} = 50$) and in less malicious environments (higher levels of honest reporters and good sellers), as we expected. With a higher number of honest raters in the system, the false negatives however could not be improved as significantly as the false positives. This might be due to our nonoptimized implementation of the PeerTrust approach to estimate a rating's reliability, e.g., it did not learn from the previous false negatives in a noisy environment where the good sellers might occasionally provide bad services. It is expected that in practice optimized implementations of such a model would be much more resilient and yield significantly lower false positive and false negative rates. Nevertheless, even in the worst case of the simulation, $\max\{\alpha, \beta\}$ was well bellow 0.4, and thus there existed a threshold k for which Theorem 3.2 was applicable (Figure 1 requires $k \geq 2$ in this case). Since it is more interesting to consider the more representative cases with all different types of sellers, we mainly focused on the average cases C_2 and C_3, where it may be approximated that $\max\{\alpha, \beta\} < \varepsilon = 0.25$. This error bound ε well satisfied the maximal error bound ε_{max} required by Theorem 3.2, with any threshold $k \geq 1$ (c.f. Figure 1). These $max\{\alpha, \beta\}$ statistics were then used in our later experiments as the global knowledge ε of all peers. We also observed the same trend of these accuracy statistics in the presence of strategic peers in later experiments. Different values of μ_{trans} were also tested. We observed that when most sellers participated in a few transactions ($\mu_{trans} < 10$), the computational trust model did not have sufficient data for its learning, resulting in high $\alpha, \beta > 0.5$, and our peer selection approach using the last rating was ineffective in enforcing cooperation.

6.5. Cooperation in Various Environments

The levels of cooperation in the system with different types of sellers and raters are given in Figure 8 for $\mu_{trans} = 50$ (cases C_4, C_5, C_6 of Table II). The case of all strategic sellers showed even a better trend. Experiments were also performed in other extreme cases, e.g., all sellers, buyers, and raters were strategic and gave similar results. For small $\mu_{trans} < 10$ the cooperation dropped significantly, as the evaluation of rating was not reliable. Given many types of rating and service behaviors (as in the cases C_5, C_6 of Table II), the accumulated utilities of strategic peers who followed the serving



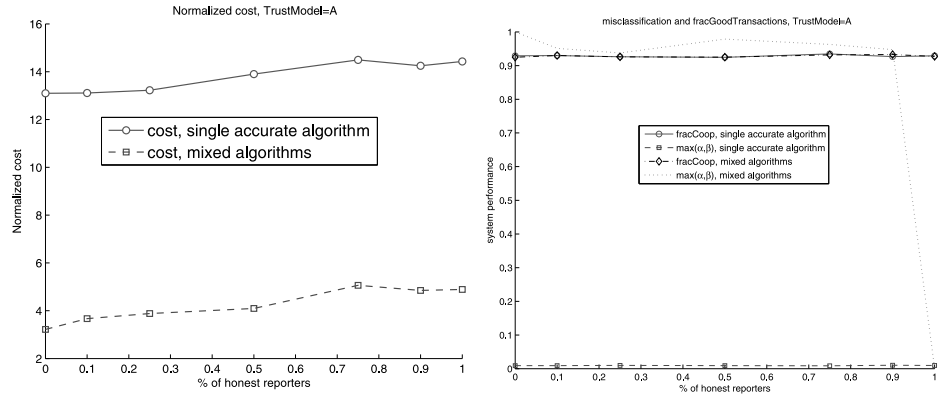
(a) Fraction of good transactions (b) Number of good transactions per peer

Fig. 9. Cooperation in the system with different ε and k (the cases C_4^k, C_6^k of Table II).

strategies designed by our theoretical analysis were also measured (not shown). The utilities of the strategic sellers were close to those of the good ones and significantly higher than the bad sellers. The reason was that strategic peers were enforced to cooperate in most of their transactions. In fact, fully cooperative sellers had slightly better utilities. This was due to our pessimistic implementation of strategic peers, who did not cooperate during their last Δ transactions. What happened was that in the last Δ transaction, several strategic peers who ever cheated were also blacklisted. Therefore, under the proposed peer selection mechanism it would be better off for a strategic seller to cooperate even in its last transaction. The proposed approach still worked even with dynamic joins and leaves of peers in the system, given that most sellers stayed in the system long enough ($\mu_{trans} > 10$ in our simulation).

The impact of threshold k is shown in Figure 9 for the cases C_4^k, C_6^k. Note that the measurement in Figure 9(b) was the accumulated number of good transactions per peer, which increased over time and whose absolute values were less important. We sampled it at the time where the simulation reached the stationary regime only to compare the difference between the number of good transactions completed under different peer-selection schemes with various k. We made the following observations. First, for a given ε, a lower k led to a higher fraction of cooperation in the system (Figure 9 a), yet with a lower accumulative number of good transactions (Figure 9b). The reason was that with lower k and higher ε, the buyers wrongly blacklisted some good sellers and thus the total number of good transactions was smaller (though the percentage of good transactions was still high). Second, for a small $\varepsilon = 0.25$, the threshold k could be increased to increase the transaction rate while not reducing the percentage of good transactions, e.g., the case $C_6^k, \epsilon = 0.25, k = 1, 2, 5$.

In the case of all strategic peers C_7, we also used a combination of two computational trust models to measure the save in the total communication cost (Theorem 4.1). Figure 10 shows the cost of two approaches. The first used only the hypothetically accurate yet expensive trust model $\mathcal{A}$, while the second used $\mathcal{A}$ with only a low probability. Therein, the cost of a computational trust model was measured as the average number of sent and received messages for a good transaction. It is interesting to see that the communication cost could be reduced significantly by using the expensive model with a very low frequency while still maintaining a high level of cooperation in the system. Furthermore, the approach combining two trust models learned faster, and during the same simulation time, the total number of good transactions in the system (not shown in Figure 10) was also much higher.



(a) Cost of the expensive trust model $\mathcal{A}$ and of the approach mixing $\mathcal{A}$ with the naive model $\mathcal{N}$

(b) Misclassification errors and cooperation

Fig. 10. A comparison of the communication cost between two approaches: the use of a single accurate, expensive algorithm $\mathcal{A}$, and the mix of $\mathcal{A}$ and the naive algorithm $\mathcal{N}$.

7. RELATED WORK

Several existing works focus on the development of various computational models to learn peers' trustworthy behavior based on their historical performance [Despotovic and Aberer 2006; Golbeck 2006; Jøsang et al. 2007]. These are complementary to our work, since any robust and accurate computational trust models can be used in our reputation-based peer selection protocol seamlessly.

Our work is inspired by the analysis of Dellarocas [2005a], which studies various design parameters of reputation mechanisms. Some conclusions in Dellarocas [2005a] coincide with the analysis in this article, e.g., the robustness of the naive trust model algorithm. However, our work considers many aspects of a reputation-based computational trust model, namely its accuracy and cost, as well as the possibility of using such a model to effectively enforce cooperation among providers in heterogenous environments with rational and malicious participants.

Our cost-efficient reputation management approach in Section 4 is an application of inspection game theory [Avenhaus et al. 2002], where an accurate and expensive computational trust model plays the role of an inspector detecting the dishonest behavior of a provider (an inspectee). The provided result confirms the important role of an effective identity management scheme, as earlier identified by Friedman and Resnick [2001]. The work most related to our approach is Agrawal and Terzi [2006], yet it addresses another problem of how to control the behaviors of agents in a centralized sovereign information-sharing scenario. Our work studies the possibility of using computational trust mechanisms in a cost-effective way and provides general results applicable to a wider range of applications with different degrees of centralization.

8. CONCLUSION

We have presented a theoretical analysis and extensive simulation to study the possibility of using a reputation-based computational trust model to enforce cooperation in a heterogenous environment where rational and malicious behaviors are present. We have analyzed a peer selection protocol in which a client uses a trust model to evaluate the reliability of the most recent rating on a provider, so as to select the most eligible one for its next transaction. Such a simple selection protocol effectively filters out irrationally malicious providers and creates a social control mechanism to stimulate cooperation of rational providers in most of their transactions. To a larger extent, the

presented protocol establishes an umbrella framework to use reputation information effectively by exploiting both its sanctioning and signaling roles [Dellarocas 2005a] in decentralized and self-organized systems.

The theoretical result of this article proves that the key to ensuring cooperation is not the accuracy/errors of the trust learning algorithm being used, but the effectiveness of identity management. Establishing a new identity must be costly so that the rational peers want to stay in the system for many transactions rather than changing their identities and starting over. Such whitewashing behaviors can be prevented by using available techniques, for example, to impose a direct entrant fee on newly joined peers, or to give an identity premium to each participant such that keeping the same identity is its best strategy [Vu and Aberer 2010].

Many interesting implications can also be derived from our work. First, any existing computational trust model with reasonable misclassification errors in identifying dishonest ratings can be used effectively to enforce cooperation in a heterogeneous environment with malicious and rational behaviors. Second, our analysis implies that when peers use different algorithms with a certain acceptable accuracy to learn the trustworthiness of their potential partners, cooperation still emerges. Third, depending on the presence of rationality from participants of the scenario being studied, either simple or sophisticated trust learning algorithms may be appropriate. In environments with long-term and fully rational peers wanting to maximize their utilities, any trust model with a bounded error is sufficient to stimulate cooperation. In the scenarios where malicious peers want to attack the system at any cost, sophisticated algorithms are necessary to filter out malicious participants.

As future work, it is of our interest to derive some theoretical bounds on the cooperation level in the system, using the proposed reputation management approach, given the arrival and up-time distributions of peers. An implementation of various serving strategies of bounded rational peers in the simulation framework may be also necessary. Such empirical simulations may give us more insights into the effectiveness of different trust learning algorithms in enforcing cooperation in the presence of bounded-rational peers with limited reasoning capability. At the end of the day, this combination of an analytical and a simulation-based approach would help us to provide a "cookbook" for the design and application of targeted, cost-effective, and realistic decentralized trust management approaches, such as needed for peer-to-peer, electronic commerce, or community systems.

ELECTRONIC APPENDIX

Due to space limitations, all proofs are in an electronic appendix that can be accessed in the ACM Digital Library.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their detailed comments on the revision of this manuscript.

REFERENCES

ABERER, K. AND DESPOTOVIC, Z. 2001. Managing trust in a peer-2-peer information system. In *Proceedings of the 10th International Conference on Information and Knowledge Management (CIKM)*, 310–317.

ABERER, K., CUDRÉ-MAUROUX, P., DATTA, A., DESPOTOVIC, Z., HAUSWIRTH, M., PUNCEVA, M., AND SCHMIDT, R. 2003. P-Grid: A self-organizing structured P2P system. *SIGMOD Rec. 32,* 3, 29–33.

AGRAWAL, R. AND TERZI, E. 2006. On honesty in sovereign information sharing. In *Proceedings of the International Conference on Extending Database Technology (EDBT)*. Lecture Notes in Computer Science, vol. 3896. Springer, 240–256.

ANCEAUME, E. AND RAVOAJA, A. 2006. Incentive-based robust reputation mechanism for p2p services. In *Proceedings of the International Conference On Principles Of Distributed Systems (OPODIS)*. 305–319.

ASHRI, R., RAMCHURN, S. D., SABATER, J., LUCK, M., AND JENNINGS, N. R. 2005. Trust evaluation through relationship analysis. In *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent systems (AAMAS)*. 1005–1011.

AVENHAUS, R., STENGEL, B. V., AND ZAMIR, S. 2002. Inspection games. In *Handbook of Game Theory with Economic Applications 3*, 1947–1987.

BUYYA, R., STOCKINGER, H., GIDDY, J., AND ABRAMSON, D. 2001. Economic models for management of resources in peer-to-peer and grid computing. In *Proceedings of the SPIE International Conference on Commercial Applications for High-Performance Computing*.

CORNELLI, F., DAMIANI, E., VIMERCATI, S. C., PARABOSCHI, S., AND SAMARATI, P. 2002. Choosing reputable servents in a P2P network. In *Proceedings of the 11th International Conference on World Wide Web (WWW)*. ACM Press, New York, 376–386.

DATTA, A., HAUSWIRTH, M., AND ABERER, K. 2003. Beyond "web of trust": enabling p2p e-commerce. In *Proceedings of the IEEE International Conference on E-Commerce (CEC)*. 303–312.

DELLAROCAS, C. 2005a. Reputation mechanism design in online trading environments with pure moral hazard. *Inform. Syst. Res. 16*, 2, 209–230.

DELLAROCAS, C. 2005b. Reputation Mechanisms. In *Handbook on Economics and Information Systems*, T. Hendershott, Ed., Elsevier.

DESPOTOVIC, Z. 2005. Building trust-aware P2P systems. Ph.D. thesis, Swiss Federal Institute of Technology Lausanne.

DESPOTOVIC, Z. AND ABERER, K. 2004. A probabilistic approach to predict peers' performance in P2P networks. In *Proceedings of the 8th International Workshop on Cooperative Information Agents VIII (CIA)*. M. Klusch, S. Ossowski, V. Kashyap, and R. Unland, Eds., Lecture Notes in Computer Science, vol. 3191. 62–76.

DESPOTOVIC, Z. AND ABERER, K. 2006. P2P reputation management: Probabilistic estimation vs. social networks. *J. Comput. Netw., (Special Issue on Management in Peer-to-Peer Systems: Trust, Reputation and Security) 50*, 4, 485–500.

FRIEDMAN, E. J. AND RESNICK, P. 2001. The social cost of cheap pseudonyms. *J. Econ. Manage. Strat. 10*, 2, 173–199.

GOLBECK, J. 2006. Trust on the World Wide Web: A survey. *Found. Trends in Web Sci. 1*, 2, 131–197.

GRAHAM, P. 2002. *A Plan For Spam, Hackers and Painters, Big Ideas from the Computer Age*. O'Really.

GUMMADI, K. P., SAROIU, S., AND GRIBBLE, S. D. 2002. King: Estimating latency between arbitrary internet end hosts. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment (IMW)*. 5–18.

JØSANG, A., ISMAIL, R., AND BOYD, C. 2007. A survey of trust and reputation systems for online service provision. *Decis. Support Syst. 43*, 2, 618–644.

JURCA, R. AND FALTINGS, B. 2006. Minimum payments that reward honest reputation feedback. In *Proceedings of the ACM Conference on Electronic Commerce*. 190–199.

KAMVAR, S. D., SCHLOSSER, M. T., AND MOLINA, H. G. 2003. The EigenTrust algorithm for reputation management in P2P networks. In *Proceedings of the International Conference on World Wide Web (WWW)*.

LEVIEN, R. 2002. Attack-resistant trust metrics. Ph.D. thesis, University of California at Berkeley.

MILLER, N., RESNICK, P., AND ZECKHAUSER, R. 2005. Eliciting informative feedback: The peer-prediction method. *Manage. Sci. 51*, 9, 1359–1373.

NECULA, G. C. 1997. Proof-carrying code. In *Proceedings of the 24th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*. ACM, New York, 106–119.

NORTH, M. J., COLLIER, N. T., AND VOS, J. R. 2006. Experiences creating three implementations of the repast agent modeling toolkit. *ACM Trans. Model. Comput. Simul. 16*, 1, 1–25.

PAPAZOGLOU, M. P. AND GEORGAKOPOULOS, D. 2003. Service-oriented computing. *Commun. ACM 46*, 10, 24–28.

RESNICK, P., KUWABARA, K., ZECKHAUSER, R., AND FRIEDMAN, E. 2000. Reputation systems. *Comm. ACM 43*, 12, 45–48.

STUTZBACH, D. AND REJAIE, R. 2006. Understanding churn in peer-to-peer networks. In *Proceedings of the 6th ACM SIGCOMM on Internet Measurement (IMC)*. ACM Press, New York, NY, 189–202.

SUN, Y. L., HAN, Z., YU, W., AND LIU, K. J. R. 2006. A trust evaluation framework in distributed networks: Vulnerability analysis and defense against attacks. In *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*.

VU, L.-H. AND ABERER, K. 2007. A probabilistic framework for decentralized management of trust and quality. In *Proceedings of the International Workshop on Cooperative Information Agents (CIA)*. 328–342.

VU, L.-H. AND ABERER, K. 2008. Effective usage of computational trust models in rational environments. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 583–586.

VU, L.-H. AND ABERER, K. 2010. Using identity premium for cooperation enforcement and whitewashing prevention in rational environments. *EPFL Tech. Report*. https://infoscience.epfl.ch/record/150847.

XIONG, L. AND LIU, L. 2004. PeerTrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Trans. Knowl. Data Eng. 16*, 7, 843–857.

Received August 2008; revised April 2009, September 2009; accepted August 2010